Project 0.1: (in Java) This project is for you to practices on a simple I/O in Java via the implementation of the basic binary threshold operations in Java:

The binary threshold: given an image and a threshold value, the binary threshold operation is to transform pixels in the input image from grey-scale to binary values, where

```
if imgIn(i, j) >= threshold value then
        result(i, j) ← 1
else
        result(i, j) ← 0
```

// You should be able do this in a hour or two.

****

Language: Java

****

Project points: 2 pts

Due Date:

-0 (2/2 pts): on time, 1/31/2023 Tuesday before midnight

-1 (1/2 pts): 1 day late: 2/1/2023 Wednesday before midnight

(-2/2 pts): non-submission, 2/2/2023, Wednesday after midnight

*** Name your soft copy and hard copy files using the naming convention (include in the email).

*** All on-line submission MUST include Soft copy (*.zip) and hard copy (*.pdf) in **the same email attachments** with correct email subject as stated in the project submission requirement; otherwise, your will be rejected. submission

****

I. Inputs:

a) inFile1 (args[0]): a txt file representing a grey-scale image, where the first text line (4 integers) is the "header" of the input image then follows by rows and cols of integers. The header of an image consists of four integers:

numRows – number of rows in the input image

numCols – number of columns in the input image

minVal – the minimum grey scale value in the input image

MaxVal – the maximum grey scale value in the input imge

For example,

```
4  6  1 12       // image has 4 rows, 6 cols, min is 1, max is 12
2  3  4 11  2  9
5  6 11  2 10  7
1  1 12  1  9  9
4  5  6  9  9  9
```

b) Console input: ask the user for a threshold value

// for this project use threshold value 6.

****

II. a) outFile1(args[1]): The result of binary threshold of input image.

Note: The output binary image also needs to have the image header.

For example, given the above image and 6 as the threshold value then the binary image would be:

```
4  6  0  1                    // notice the min and max values have changed!
0  0  0  1  0  1
0  1  1  0  1  1
0  0  1  0  1  1
0  0  1  1  1  1
```

****
III. Data structure:
****

- numRows (int)
- numCols (int)
- minVal (int)
- maxVal (int)
- thrValue (int)

Method:

Processing (…)

****
III. Main
****

step 0: inFile1 ← open args[0]
outFile1 ← open args[1]
step 1: numRows, numCols, minVal, maxVal ←read from inFile
step 2: thrValue ← ask user from console
step 3: outFile1 ← write numRows, numCols, 0, 1 to outFile1
step 4: processing (inFile1, outFile1, thrValue)
step 5: close all files

****
IV. processing (inFile1, outFile1, thrVal)
****

step 0: (int) pixelVal
step 1: pixelVal ← read one integer from inFile
step 2: if pixelVal >= thrVal
outFile1 ←write 1 follows by 1 blank to outFile1
else
outFile1 ← write 0 follows by 1 blank to outFile1

step 3: repeat step 1 - step 2 until the inFile1 is empty